

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 11-134203

(43)Date of publication of application : 21.05.1999

(51)Int.Cl.

G06F 9/46

G06T 1/00

H04N 7/24

(21)Application number : 10-238057

(71)Applicant : MATSUSHITA ELECTRIC IND CO LTD

(22)Date of filing : 25.08.1998

(72)Inventor : IMANISHI HIROSHI

ARAKI TOSHIYUKI

(30)Priority

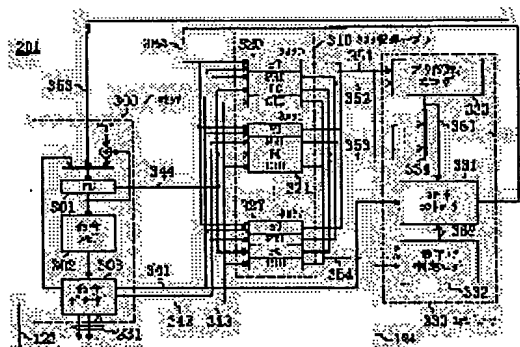
Priority number : 09235625 Priority date : 01.09.1997 Priority country : JP

(54) MICRO CONTROLLER, DATA PROCESSING SYSTEM AND CONTROL METHOD FOR TASK SWITCH

(57)Abstract:

PROBLEM TO BE SOLVED: To realize high speed task switching in a micro controller for controlling plural hardware engines.

SOLUTION: A processor 300, a task management table 310 and a scheduler 330 are incorporated in the micro controller. The processor 300 sequentially executes plural tasks for controlling the allocated hardware engines (cores). The task management table 310 stores task management information containing state information (ST information) showing the respective execution situations of the plural tasks, priority information (PRI information) showing the execution priority of the plural tasks and core ID information (CID information) showing to which cores the plural tasks are allocated. The scheduler 330 causes the processor 300 to switch the task based on task management information when a specified instruction is decoded or the execution of any core terminates.



LEGAL STATUS

[Date of request for examination] 25.08.1998

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number] 3007612

[Date of registration] 26.11.1999

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

THIS PAGE BLANK (USPTO)

THIS PAGE BLANK (USPTO)

THIS PAGE BLANK (USPTO)

THIS PAGE BLANK (USPTO)

【請求項14】 少なくとも1つのタスクがそれぞれ対応するハードウェアエンジンに割り当てられ、その割り当てを表す情報に基づいてタスクエンジンでタスクスイッチを制御するタスクスイッチの制御方法であって、各タスクの処理をその内容に応じて、プロセッサによりスケジュールされる部分と、複数のハードウェアエンジンにより実行される部分とに分割し、各ハードウェアエンジンは、前記プロセッサによる分割に従ってデータ処理の実行を開始し、かつ該データ処理の実行が終了すると前記タスクエンジンに対して実行終了を知らせる。

各タスクは、実行待ちを表す第1のステートと、実行中を表す第2のステートと、割り当てられたハードウェアエンジンでの実行終了待ちを表す第3のステートとを有し、前記タスクエンジンは、いずれかのハードウェアエンジンが実行終了したとき、前記プロセッサにタスクスイッチをさせるように、実行終了したハードウェアエンジンに割り当てられたタスクのステートを前記第3のステートから前記第1のステートへ変更することを特徴とするタスクスイッチの制御方法。

【請求項15】 請求項14記載のタスクスイッチの制御方法において、ハードウェアエンジンが実行終了したとき、実行中のタスクのステートを前記第2のステートから前記第1のステートへ変更することを特徴とするタスクスイッチの制御方法。

【請求項16】 複数のハードウェアエンジンと接続して、かつプログラムされた命令に従って、複数のタスクをスケジュールに実行するためのプロセッサと、前記複数のタスクの各々の実行状況を表すステート情報と、前記複数のタスクの各々の実行優先度を表すプライオリティ情報と、前記複数のタスクの各々がいずれのハードウェアエンジンに割り当てられているかを表す割り当て情報とを含むタスク管理情報を記憶するためのタスク管理テーブルと、前記タスク管理情報に基づいて前記プロセッサにタスクスイッチをさせるためのスケジュールと、前記複数のハードウェアエンジンに共通の既定パラメータを記憶するためのレジスタテーブルとを備えたことを特徴とするマイクロコントローラ。

【発明の詳細な説明】
【0001】
【発明の属する技術分野】 本発明は、マルチタスク環境を備えたマイクロコントローラと、該マイクロコントローラが複数のハードウェアエンジンに割り付けられるように構成されたデータ処理システムとに関するものである。また、本発明はタスクスイッチの制御方法に関するものである。

【0002】
【従来の技術】 マルチタスク環境を備えたマイクロコントローラが知られている。この種のマイクロコントローラに内蔵された単一のプロセッサは、複数のタスクをスケジュールに実行する。そのため、タスクスイッチを要求するタイマ割り込みをタスクスイッチが定期的に発行する。このタイマ割り込みをプロセッサが定期的に発行し、オペレーティングシステム (OS) 内の割り込み処理ルーチンが起動され、該割り込み処理ルーチンがタスクスケジュールにリソースの遅延及び復帰を行なう。

【0003】
【発明が解決しようとする課題】 上記従来のマイクロコントローラは、割り込み処理ルーチンでタスクスケジュールを行っていたことから、タスクスイッチにおけるオーバーヘッドが大きくなり、マイクロコントローラの実質的な稼働率が低下するという問題があった。これは、特に画像データのエンコードのようなリアルタイム性を重視するアプリケーションにおいて深刻な問題である。

【0004】 本発明の目的は、マイクロコントローラにおける高速のタスクスイッチを実現することにある。

【0005】 本発明の他の目的は、マイクロコントローラが複数のハードウェアエンジンに割り付けられるように構成されたデータ処理システムにおいて、該マイクロコントローラにおける高速のタスクスイッチを実現することにある。

【0006】 本発明の更に他の目的は、高速のタスクスイッチを実現するためのタスクスイッチの制御方法を提供することにある。

【0007】

【課題を解決するための手段】 上記目的を達成するため、本発明のマイクロコントローラは、割り込み処理ルーチンでタスクスイッチを制御するのではなく、当該マイクロコントローラ中のプロセッサが複数のハードウェアエンジンと協調してかつプログラムされた命令に従って複数のタスクをスケジュールに実行し、かつ複数のタスクがそれぞれ対応するハードウェアエンジンに割り当てられた状態で、その割り当てを表す情報に基づいてハードウェアスケジュールでタスクスイッチを制御することとしたものである。複数のハードウェアエンジンの各々は、プロセッサによる起動に従ってデータ処理の実行を開始し、かつ該データ処理の実行が終了するとタスクエンジンに対して実行終了を知らせる。スケジュールは、いずれかのハードウェアエンジンの実行終了を待った場合には、プロセッサにタスクスイッチをさせる。スケジュールは、いずれかのハードウェアエンジンの実行終了を待った場合には、プロセッサにタスクスイッチをさせる。しかも、複数のタスクの処理を、その内容に応じて、プロセッサによりスケジュールに実行される部分と、複数のハードウェアエンジンにより実行される部分とに分割する構成とした。複数のハードウェアエンジンの中には時間的にクリティカルな処理を実行するもの

と、そうでないものがある。本発明によれば、このような複数のハードウェアエンジン間の関係が複数のタスクの各々の実行優先度に応じて結果、いずれかのハードウェアエンジンが時間的にクリティカルな処理を実行するのをタスクスイッチ時にあらかじめ判断することなく、次に実行すべきタスクを短時間のうちに選択できる。つまり、タスクスイッチにおけるオーバーヘッドが小さくなり、高速のタスクスイッチが実現される。

【0008】 また、本発明のマイクロコントローラでは、タスクスイッチから定期的に発行される割り込みに応じてタスクスイッチを行う時分方式では無駄な時間が生じることになり、各ハードウェアエンジンの実行終了というイベントの発生に直ちに反応してタスクスイッチを行うイベントトリガ方式を採用した。複数のタスクの各々は、実行待ちを表す第1のステート (READYステート) と、実行中を表す第2のステート (ACTIVEステート) と、割り当てられたハードウェアエンジンにおける実行終了待ちを表す第3のステート (SLEEPステート) とを有する。ACTIVEステートは、現在、タスクがマイクロコントローラを使用している状態であり、そのタスクに割り当てられたハードウェアエンジンの制御を行う。READYステートは、タスクがマイクロコントローラを使用可能な状態ではあるが、そのタスクは選ばれておらず、選ばれたのを待っている状態である。SLEEPステートは、割り当てられたハードウェアエンジンの起動を待っている状態である。割り当てられたハードウェアエンジンの起動を終えたタスクは、特定の命令 (task_sleep命令) に応じて、ACTIVEステートからSLEEPステートへ移行する。あるハードウェアエンジンが実行終了したとき、該ハードウェアエンジンに割り当てられたタスクはSLEEPステートからREADYステートへ移行し、実行中のタスクのうち最も高い実行優先度を有するタスクが次に実行すべきタスクとして選択され、該選択されたタスクがREADYステートからACTIVEステートへ移行する。

【0009】 複数のハードウェアエンジンの各々が互いに独立した作業領域として使用する複数のレジスタファイルとマイクロコントローラの中に用意しておけば、タスクスイッチ時にはプログラムカウンタなどのプロセッサ資源のみを迅速にばくく、タスクスイッチにおけるオーバーヘッドが更に小さくなる。複数のハードウェアエンジンに共通の既定パラメータを記憶するためのレジスタファイルとマイクロコントローラの中に用意しておいてよい。

【0010】

【発明の実施の形態】 図1は、本発明に係るデータ処理

システムの1つであるMPPEG (Moving Picture Experts Group) 画像エンコーダの構成例を示している。図1のエンコーダは、単一のマイクロコントローラ101と、マイクロコントローラ101を構成する5個のハードウェアエンジン (以下、コアという) 111~115と、3個のバッファメモリ116~118とで構成されている。5個のコアは、動き検出器 (Motion Detector: MD) 111、動き補償器 (Motion Compensator: MC) 112、離散コサイン変換器 (Discrete Cosine Transformer: DCT) 113、量子化器 (Quantizer: Q) 114及び可変長符号化器 (Variable Length Code: VLC) 115であって、マルチタスク環境を備えたマイクロコントローラ101によりそれぞれ制御される。121はエンコードされるべき画像データ、122はエンコード結果を表す符号化データである。マイクロコントローラ101は、5個のコア111~115の各々へ起動信号123を供給し、かつ5個のコア111~115の各々から終了信号124を受け取る。また、マイクロコントローラ101は、信号線131~135を介して5個のコア111~115と個別に通信したり、信号線136を介して5個のコア111~115に共通のパラメータを与えたりすることができるように

なっている。

【0011】 図1は、マイクロコントローラ101の詳細構成を示している。マイクロコントローラ101は、マルチタスク環境を実現するためのタスクコントローラ201と、上記5個のコア111~115の各々が互いに独立した作業領域として使用する5個のコアレジスタファイル211~215と、上記5個のコア111~115の各々の少なくとも2個のコアに共通の既定パラメータを記憶するための1個の共通レジスタファイル216と、タスクコントローラ201が作業領域として使用する1個の汎用レジスタファイル217と、乗算器221と、シフト222と、算術論理演算ユニット (Arithmetic and Logic Unit: ALU) 223と、データメモリ224とを備えている。241はAバス、242はBバス、243はCバス、231はこれらのバスとタスクコントローラ201とを接続するための信号線である。タスクコントローラ201は、上記起動信号123を供給し、かつ上記終了信号124を受け取る。レジスタファイル211~216の各々は、Cバス243と上記記号線131~136のうちの対応する信号線との間に介在し、かつその各々の2出力がAバス241及びBバス242にそれぞれ接続されている。乗算器221、シフト222及びALU223の各々の出力はAバス241及びBバス242にそれぞれ接続されている。なお、5個のコアレジ

9

スタタアル211～215及び共通レジスタアル216の配線を省略して、上記信号線131～136をCバス243から直接に引き出すようにしてもよい。

【0012】図1のMP EG画像エンコーダによれば、16×16画素からなるマクロブロックを単位として画像データ処理が進められる。まず、入力された画像データ121に対してMDコプ111で動きベクトルの検出が求められる。これらの動きベクトルを用いてMCコプ112で画像の差分データが求められ、最適な動きベクトルが選択される。選択された動きベクトルに対する差分データは、DCTコプ113で離散コサイン変換され、Qコプ114で量子化され、求めた動きベクトルなどのサイド情報とともにVLCコプ115で可変長符号化された後、符号化データ122として出力される。

【0013】図2を参照して詳細に説明すると、タスクコントローラ201は、まず信号線231、ALU223及びCバス243を介してMDコプレジスタアル211に動作パラメータを設定し、起動信号123によりMDコプ111を起動する。MDコプ111は、MDコプレジスタアル211から信号線131を介して動作パラメータを読み込むとともに、画像データ121を入力する。MDコプ111の実行が終了すると、求められた動きベクトルの検出が信号線131を介してMDコプレジスタアル211に書き込まれ、MDコプ111から終了信号124が出力される。タスクコントローラ201は、この終了信号124を受け取り、MDコプレジスタアル211から動きベクトルの検出を読み出し、これに基づき乗算器221、シフト222、ALU223及び汎用レジスタアル217を用いてMCコプ112のための動作パラメータを計算する。この動作パラメータはMCコプレジスタアル212に設定され、起動信号123によりMCコプ112が起動される。MCコプ112は、MCコプレジスタアル212から信号線132を介して動作パラメータを読み込み、その後、画像の差分データを求める。MCコプ112の実行が終了すると、差分データの検出が信号線132を介してMCコプレジスタアル212に、画像の差分データがバスメモリ116にそれぞれ書き込まれ、MCコプ112から終了信号124が出力される。タスクコントローラ201は、この終了信号124を受け取り、MCコプレジスタアル212から差分データの検出を読み出し、これに基づき乗算器221、シフト222、ALU223及び汎用レジスタアル217を用いて上記乗算の中から最適な動きベクトルを選択する。求められた動きベクトルに対する差分データの検出がDCTコプレジスタアル213に設定される。DCTコプ113は、DCTコプレジスタアル213に設定されたアルゴリズムに基づいてバスメモリ116から差分データを読み出し、これを離散コサイン

10

変換する。DCTコプ113の実行が終了すると、離散コサイン変換の検出がバスメモリ117に書き込まれ、DCTコプ113から終了信号124が出力される。以下、Qコプ114で量子化処理がなされ、その結果がバスメモリ118に書き込まれ、VLCコプ115で可変長符号化処理がなされる。この結果が符号化データ122として出力される。なお、上記5個のコプ111～115のうちいくつかのコプは、マクロブロックを処理する間に、マクロコントローラ101と起動信号123及び終了信号124のやりとりを複数回行う。共通レジスタアル216は、MP EG1とMP EG2との切り替えるための共通パラメータを5個のコプ111～115へ予め供給したり、動き評価モードを指定するための共通パラメータをMDコプ111及びMCコプ112へ予め供給したりする際に用いられる。

【0014】図3は、タスクコントローラ201の詳細構成を示している。タスクコントローラ201は、プロセッサ300と、タスク管理テーブル310と、スタジアム300とで構成されている。プロセッサ300は、最大8個のタスクをスケジューリングに実行することができるRISC (Reduced Instruction Set Computer) 型のプロセッサである。命令アドレスを生成するためのプログラムカウンタ (PC) 301と、一連の命令からなるプログラムを格納するための命令メモリ302と、命令をフェッチするための命令デコーダ303とを備えている。各コプへの起動信号123は、命令デコーダ303から供給される。また、命令デコーダ303は、命令実行のための資源である上記乗算器221、シフト222、ALU223などに信号線231を介して接続されている。タスク管理テーブル310は、タスク管理情報を記憶するための回路ブロックであって、タスク40からタスク7までの8個のタスクのそれぞれに対応した8個のタスク管理レジスタアル320～327を備えている。ここに、タスク管理情報は、複数のタスクの各々の実行状況を表すステータス情報 (ST情報) と、該複数のタスクの各々の実行優先度を表すプライオリティ情報 (PRI情報) と、該複数のタスクの各々の5個のコプ111～115のうちいずれのコプに割り当てられているかを表すコプID情報 (CID情報) とを含む。更に、タスク管理テーブル310は、プロセッサ300の資源の領域を有するFC301の内容を返還するためのタスク毎の領域を含む。この返還領域は、ALU223 (図2参照) の演算結果に係るラックなども含まれる。スケジューラ330は、タスク管理テーブル310に記憶されたタスク管理情報に基づいてプロセッサ300にタスクアイコンをさせるための回路ブロックであって、スケートコントローラ331と、終了コプ判定ユニット332と、プライオリティエンコーダ333と、セレクトカ334とを備えている。終了コプ判定ユニット332は、5個のコプ111～115のうちいずれ

11

れのコプから終了信号124を受け取ったとき、実行終了したコプに割り当てられたタスクを判定するためのユニットである。この判定はタスク管理テーブル310を参照して行われ、判定結果を表すタスク番号362がステートコントローラ331に知らされる。プライオリティエンコーダ333は、次に実行すべきタスクを選択するための回路ブロックである。この選択はタスク管理テーブル310を参照して行われ、選択結果を表すタスク番号361がステートコントローラ331及びセレクトカ334に知らされる。ステートコントローラ331は、タスク管理テーブル310の中のST情報を更新するための回路ブロックである。セレクトカ334は、プロセッサ300への資源渡与を行う。

【0015】図4は、図1のMP EG画像エンコーダにおけるコプとタスクとの対応関係を示している。マクロコントローラ101は、ここでは6個のタスク400～405を実行する。タスク400は、下位階層の5個のタスク401～405を制御し、かつエンコード処理の全体を管理するためのメインタスク (タスク40) である。このメインタスク400が割り当てられるべきコプ111を制御するための動き検出タスク (タスク112) を制御する。タスク402は、割り当てられたMCコプ112を制御するための動き補償タスク (タスク2) である。タスク403は、割り当てられたDCTコプ113を制御するための離散コサイン変換タスク (タスク3) である。タスク404は、割り当てられたQコプ114を制御するための量子化タスク (タスク4) である。タスク405は、割り当てられたVLCコプ115を制御するための可変長符号化タスク (タスク5) である。

【0016】ここで、図4に示す少なくとも6個のタスク400～405に係るタスク管理情報が図3中のタスク管理テーブル310に記憶されているものとする。図3によれば、PRI情報はプライオリティ設定信号342に応じて、CID情報はコプ設定信号343に応じてそれぞれ設定される。プライオリティ設定信号342は命令デコーダ303がプライオリティ設定命令をデコードした際に、コプ設定信号343は命令デコーダ303がコプ設定命令をデコードした際にそれぞれ命令デコーダ303からタスク管理テーブル310へ供給される。

【0017】図5は、各タスクのステータス遷移を示す概略図である。タスクは、停止を表すSTOPステータスと、実行待ちを表すREADYステータスと、実行中を表すACTIVEステータスと、割り当てられたコプの実行終了待ちを表すSLEEPステータスを有する。ただし、タスク40にはSLEEPステータスがない。リセット直後のタスクはSTOPステータスにある。STOPステータスにあるタスクは、task_ready命令によりREADYステータスへ遷移させられる (遷移501)。READYステータスにあるタスクは、タスクアイコンを

12

要求するイベントの発生時にスケジューラ330により選択されると、ACTIVEステータスへ遷移させられる (遷移511)。この際、その時点までACTIVEステータスにあったタスクは、スケジューラ330によりREADYステータスへ遷移させられる (遷移522)。ACTIVEステータスにあるタスクは、プロセッサ300により実行され、task_sleep命令によりSLEEPステータスへ遷移させられる (遷移521)。task_sleep命令によりSTOPステータスへ遷移させられる (遷移523) することができる。SLEEPステータスにあるタスクは、割り当てられたコプの実行終了によりREADYステータスへ遷移させられる (遷移531)。

【0018】ここで、図3のタスクコントローラ201の詳細動作を説明する。命令デコーダ303がtask_ready命令、task_sleep命令又はtask_sleep命令をフェッチすると、タスクアイコンに割り当てられたコプのための動作パラメータの設定と該コプの起動とを終えたタスクは、task_sleep命令によりACTIVEステータスからSLEEPステータスへ遷移させられる。また、5個のコプ111～115のうちいずれかのコプが実行終了したときにもタスクアイコンが発生する。タスクアイコン時のタスクコントローラ201の動作シーケンスは、(1) スケジューラの起動、(2) 実行中のタスクの資源の返還、(3) 次に実行するタスクの選択、(4) 返還されていた資源の復帰である。

【0019】まず、命令に基づきタスクアイコンのシーケンスを説明する。

【0020】(A-1) スケジューラの起動
task_ready命令、task_sleep命令又はtask_sleep命令がフェッチされると、命令デコーダ303からステータス変更信号341が出力される。ステータス変更信号341はステートコントローラ331に入力される。その結果、スケジューラ330が起動される。

【0021】(A-2) 実行中のタスクの資源の返還
ステータス変更信号341はタスク管理テーブル310にも入力され、ST情報が更新される。同時にその時点まで実行されていたタスクのFC301の値が、信号線344を介してレジスタ管理テーブル310の中に返還される。

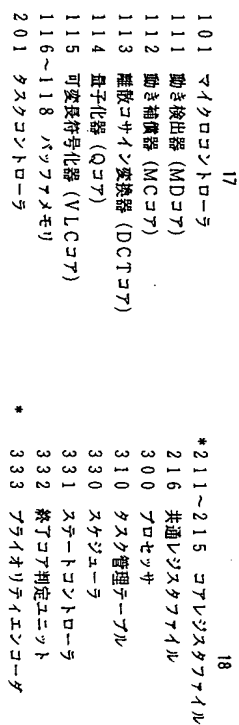
【0022】(A-3) 次に実行するタスクの選択
プライオリティエンコーダ333は、タスク管理テーブル310から信号線351を介してST情報を、信号線352を介してPRI情報をそれぞれ受け取り、READYステータスにあるタスクのうち最も高い実行優先度を有するタスクを次に実行すべきタスクとして選択する。この選択の結果を表すタスク番号361は、ステータス

50

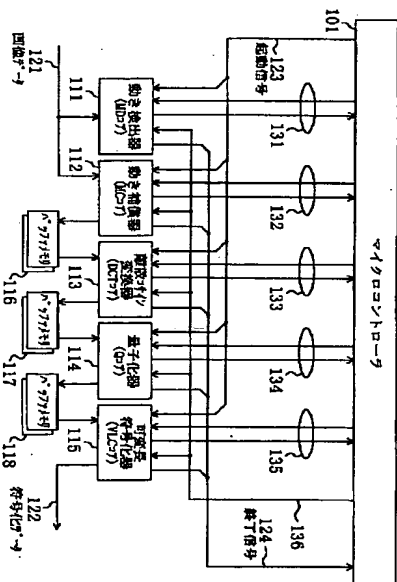
50

50

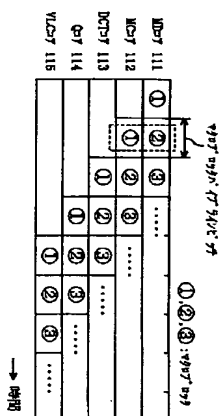
50



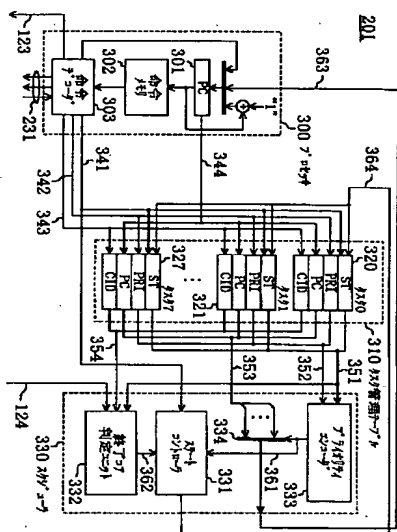
【圖 1】



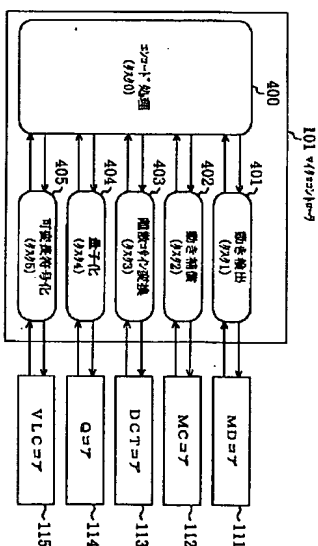
【図2】



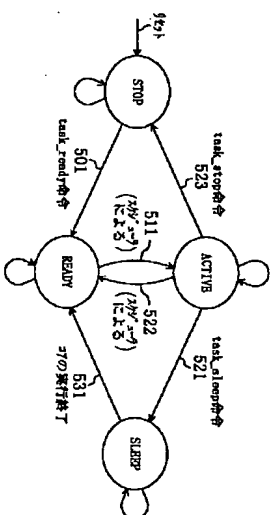
【例6】



【☒ 3】

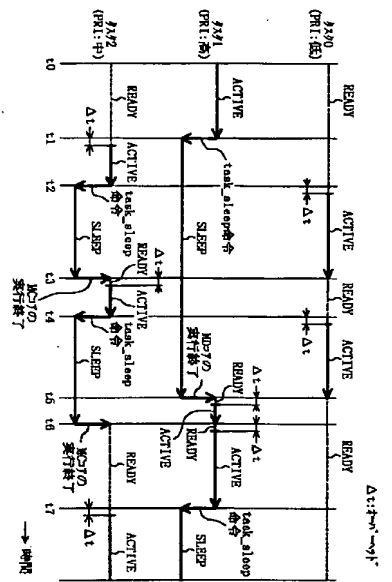


【図 4】



【例 5】

【図7】



フロントページの続き

(56) 参考文献 特開 平6-28323 (J P, A)

インターフェース1995年1月号 (CQ

出版社)、P. 134~146

National Technica

l Report, Vol. 40, No.

6 (1994年12月)、P. 122~128 (J I

C S T資料番号: G0474A)

(58) 調査した分野(Int. Cl. 7, D B 名)

G06F 9/46

G06F 15/16

G06F 9/38

G06T 1/00

H04N 7/24

J I C S Tライアル (J O I S)

C S D B (日本国特許庁)